Product Distribution Theory and Semi-Coordinate Transformations

Stéphane Airiau*
Mathematical & Computer Sciences Dept
600 South College Avenue
Tulsa, OK 74104
stephane@utulsa.edu

David H. Wolpert
NASA Ames Research Center
MS 269-2
Moffett Field, CA94035
dhw@email.arc.nasa.gov

Abstract

Product Distribution(PD) theory is a new framework for doing distributed, adaptive control of a multiagent system(MAS). We introduce the technique of "coordinate transformations" in PD theory gradient descent. These transformations selectively couple a few agents with each other into "meta-agents". Intuitively, this can be viewed as a generalization of forming binding contracts between those agents. Doing this sacri£ces a bit of the distributed nature of the MAS, in that there must now be communication from multiple agents in determining what joint-move is £nally implemented. However, as we demonstrate in computer experiments, these transformations improve the the performance of the MAS.

1. Introduction

Product Distribution (PD) theory is a recently introduced broad framework for analyzing, controlling, and optimizing distributed systems [8, 9, 10]. Among its potential applications are adaptive, distributed control of a Multi-Agent System (MAS), (constrained) optimization, sampling of high-dimensional probability densities (i.e., improvements to Metropolis sampling), density estimation, numerical integration, reinforcement learning, information-theoretic bounded rational game theory, population biology, and management theory. Some of these are investigated in [1, 2, 7].

Here we investigate PD theory's use for adaptive, distributed control of a MAS. Typically such control is done by having each agent run its own reinforcement learning algorithm [3, 11, 12, 13]. In this approach the utility function of each agent is based on the **world utility** $G(x)$ mapping the joint move of the agents, $x \in X$, to the performance of the overall system. However in practice the agents in a MAS are bounded rational. Moreover the equilibrium they reach will typically involve mixed strategies rather than pure strategies, i.e., they don't settle on a single point x optimizing $G(x)$. This suggests formulating an approach that explicitly accounts for the bounded rational, mixed strategy character of the agents.

Now in any game, bounded rational or otherwise, the agents are independent, with each agent i choosing its move x_i at any instant by sampling its probability distribution (mixed strategy) at that instant, $q_i(x_i)$. Accordingly, the distribution of the joint-moves is a product distribution, $P(x) = \prod_i q_i(x_i)$. In this representation of a MAS, all coupling between the agents occurs indirectly; it is the separate distributions of the agents $\{q_i\}$ that are statistically coupled, while the actual moves of the agents are independent.

PD theory adopts this perspective to show that the equilibrium of a MAS is the minimizer of a Lagrangian $\mathcal{L}(P)$, derived using information theory, that quanti£es the expected value of G for the joint distribution $P(x)$. From this perspective, the update rules used by the agents in RL-based systems for controlling MAS's are just particular (inef£cient) ways of £nding that minimizing distribution. PD theory suggests novel ways to £nd the equilibrium, e.g., applying any of the powerful search techniques for continuous variables, like gradient descent, to £nd the P optimizing $\mathcal{L}$. By casting the problem this way in terms of £nding an optimal P rather than £nding an optimal x, we can exploit the power of search techniques for continuous variables even when X is a discrete, £nite space.

One disadvantage of using technique such as descent is the possibility to be trapped in a local minimum. To be able to escape from a local minimum, we explore in the paper the possibility to perform a change of semi-coordinate (semi is used since the transformation needs not be invertible). To start this study, we experiments local change between two agents and study how it can produce an improvement. In the next section we review the game-theory motivation of PD theory. Then, we present the concept of semi-coordinate transformation and we present results to show that it can im-

* Student Author

prove the results signi£cantly.

2. Bounded Rational Game Theory

In this section we motivate PD theory as the information-theoretic formulation of bounded rational game theory.

2.1. Review of noncooperative game theory

In noncooperative game theory one has a set of N **players**. Each player i has its own set of allowed **pure strategies**. A **mixed strategy** is a distribution $q_i(x_i)$ over player i's possible pure strategies. Each player i also has a **private utility** function g_i that maps the pure strategies adopted by all N of the players into the real numbers. So given mixed strategies of all the players, the expected utility of player i is $E(g_i) = \int dx \ \prod_j q_j(x_j) g_i(x)$ [1].

In a **Nash equilibrium** every player adopts the mixed strategy that maximizes its expected utility, given the mixed strategies of the other players. More formally, $\forall i, q_i = \text{argmax}_{q'_i} \int dx \ q'_i \prod_{j \neq i} q_j(x_j) \ g_i(x)$. Perhaps the major objection that has been raised to the Nash equilibrium concept is its assumption of **full rationality** [4, 5]. This is the assumption that every player i can both calculate what the strategies $q_{j \neq i}$ will be and then calculate its associated optimal distribution. In other words, it is the assumption that every player will calculate the entire joint distribution $q(x) = \prod_j q_j(x_j)$. If for no other reasons than computational limitations of real humans, this assumption is essentially untenable.

2.2. Review of the maximum entropy principle

Shannon was the £rst person to realize that based on any of several separate sets of very simple desiderata, there is a unique real-valued quanti£cation of the amount of syntactic information in a distribution $P(y)$. He showed that this amount of information is (the negative of) the Shannon entropy of that distribution, $S(P) = -\int dy \ P(y) ln[\frac{P(y)}{\mu(y)}]$. So for example, the distribution with minimal information is the one that doesn't distinguish at all between the various y, i.e., the uniform distribution. Conversely, the most informative distribution is the one that speci£es a single possible y. Note that for a product distribution, entropy is additive, i.e., $S(\prod_i q_i(y_i)) = \sum_i S(q_i)$.

Say we given some incomplete prior knowledge about a distribution $P(y)$. How should one estimate $P(y)$ based on that prior knowledge? Shannon's result tells us how to do that in the most conservative way: have your estimate of $P(y)$ contain the minimal amount of extra information beyond that already contained in the prior knowledge about $P(y)$. Intuitively, this can be viewed as a version of Occam's razor. This approach is called the maximum entropy (maxent) principle. It has proven useful in domains ranging from signal processing to supervised learning [6].

2.3. Maxent Lagrangians

Much of the work on equilibrium concepts in game theory adopts the perspective of an external observer of a game. We are told something concerning the game, e.g., its utility functions, information sets, etc., and from that wish to predict what joint strategy will be followed by real-world players of the game. Say that in addition to such information, we are told the expected utilities of the players. What is our best estimate of the distribution q that generated those expected utility values? By the maxent principle, it is the distribution with maximal entropy, subject to those expectation values.

To formalize this, for simplicity assume a £nite number of players and of possible strategies for each player. To agree with the convention in other £elds, from now on we implicitly ¤ip the sign of each g_i so that the associated player i wants to minimize that function rather than maximize it. Intuitively, this ¤ipped $g_i(x)$ is the "cost" to player i when the joint-strategy is x, though we will still use the term "utility".

Then for prior knowledge that the expected utilities of the players are given by the set of values $\{\epsilon_i\}$, the maxent estimate of the associated q is given by the minimizer of the Lagrangian

$$\begin{aligned}\mathcal{L}(q) &\equiv \sum_i \beta_i [E_q(g_i) - \epsilon_i] - S(q) \\ &= \sum_i \beta_i [\int dx \ \prod_j q_j(x_j) g_i(x) - \epsilon_i] - S(q)\end{aligned}$$

where the subscript on the expectation value indicates that it evaluated under distribution q, and the $\{\beta_i\}$ are "inverse temperatures" implicitly set by the constraints on the expected utilities.

Solving, we £nd that the mixed strategies minimizing the Lagrangian are related to each other via

$$q_i(x_i) \propto e^{-E_{q_{(i)}}(G|x_i)} \tag{1}$$

where the overall proportionality constant for each i is set by normalization, and $G \equiv \sum_i \beta_i g_i$ [2]. In Eq. 1 the probability of player i choosing pure strategy x_i depends on the effect of that choice on the utilities of the other players. This

1 Throughout this paper, the integral sign is implicitly interpreted as appropriate, e.g., as Lebesgue integrals, point-sums, etc.

2 The subscript $q_{(i)}$ on the expectation value indicates that it is evaluated according the distribution $\prod_{j \neq i} q_j$.

re¤ects the fact that our prior knowledge concerns all the players equally.

If we wish to focus only on the behavior of player i, it is appropriate to modify our prior knowledge. To see how to do this, £rst consider the case of maximal prior knowledge, in which we know the actual joint-strategy of the players, and therefore all of their expected costs. For this case, trivially, the maxent principle says we should "estimate" q as that joint-strategy (it being the q with maximal entropy that is consistent with our prior knowledge). The same conclusion holds if our prior knowledge also includes the expected cost of player i.

Modify this maximal set of prior knowledge by removing from it speci£cation of player i's strategy. So our prior knowledge is the mixed strategies of all players other than i, together with player i's expected cost. We can incorporate prior knowledge of the other players' mixed strategies directly, without introducing Lagrange parameters. The resultant **maxent Lagrangian** is

$$\begin{aligned}\mathcal{L}_i(q_i) &\equiv \beta_i[\epsilon_i - E(g_i)] - S_i(q_i) \\ &= \beta_i[\epsilon_i - \int dx \prod_j q_j(x_j) g_i(x)] - S_i(q_i)\end{aligned}$$

solved by a set of coupled **Boltzmann distributions**:

$$q_i(x_i) \propto e^{-\beta_i E_{q_{(i)}}(g_i|x_i)}. \quad (2)$$

Following Nash, we can use Brouwer's £xed point theorem to establish that for any non-negative values $\{\beta\}$, there must exist at least one product distribution given by the product of these Boltzmann distributions (one term in the product for each i).

The £rst term in $\mathcal{L}_i$ is minimized by a perfectly rational player. The second term is minimized by a perfectly *irrational* player, i.e., by a perfectly uniform mixed strategy q_i. So β_i in the maxent Lagrangian explicitly speci£es the balance between the rational and irrational behavior of the player. In particular, for $\beta \to \infty$, by minimizing the Lagrangians we recover the Nash equilibria of the game. More formally, in that limit the set of q that simultaneously minimize the Lagrangians is the same as the set of delta functions about the Nash equilibria of the game. The same is true for Eq. 1.

Eq. 1 is just a special case of Eq. 2, where all player's share the same private utility, G. (Such games are known as **team games**.) This relationship re¤ects the fact that for this case, the difference between the maxent Lagrangian and the one in Eq. 1 is independent of q_i. Due to this relationship, our guarantee of the existence of a solution to the set of maxent Lagrangians implies the existence of a solution of the form Eq. 1. Typically players a will be closer to minimizing their expected cost than maximizing it. For prior knowledge consistent with such a case, the β_i are all non-negative.

For each player i de£ne

$$f_i(x, q_i(x_i)) \equiv \beta_i g_i(x) + \ln[q_i(x_i)]. \quad (3)$$

Then we can maxent Lagrangian for player i is

$$\mathcal{L}_i(q) = \int dx \; q(x) f_i(x, q_i(x_i)). \quad (4)$$

Now in a bounded rational game every player sets its strategy to minimize its Lagrangian, given the strategies of the other players. In light of Eq. 4, this means that we interpret each player in a bounded rational game as being perfectly rational for a utility that incorporates its computational cost. To do so we simply need to expand the domain of "cost functions" to include probability values as well as joint moves.

Often our prior knowledge will not consist of exact speci£cation of the expected costs of the players, even if that knowledge arises from watching the players make their moves. Such alternative kinds of prior knowledge are addressed in [9, 10]. Those references also demonstrate the extension of the formulation to allow multiple utility functions of the players, and even variable numbers of players.

3. Optimizing the Lagrangian and Algorithm

First we introduce the shorthand

$$\begin{aligned}[G \mid x_i] &\equiv E(G \mid x_i) \\ &= \int dx' \delta(x'_i - x_i) G(x) \prod_{j \neq i} q_i(x'_i),\end{aligned}$$

where the delta function forces $x'_i = x_i$ in the usual way. Now given any initial q, one may use gradient descent to search for the q optimizing $\mathcal{L}(q)$. Taking the appropriate partial derivatives, the descent direction is given by

$$\triangle q_i(x_i) = \frac{\delta \mathcal{L}}{\delta q_i(x_i)} = [G|x_i] + \beta^{-1} \log q(x_i) + C \quad (5)$$

where C is a constant set to preserve the norm of the probability distribution after update, i.e., set to ensure that

$$\int dx_i q_i(x_i) = \int dx_i (q_i(x_i) + \triangle q_i(x_i)) = 1. \quad (6)$$

Evaluating, we £nd that

$$C = -\frac{1}{\int dx_i 1} \int dx_i \Big\{ [G|x_i] + \beta^{-1} \log q_i(x_i) \Big\}. \quad (7)$$

(Note that for £nite X, those integrals are just sums.)

To follow this gradient, we need an ef£cient scheme for estimation of the conditional expected G for different x_i. Here we do this via Monte Carlo sampling, i.e., by repeatedly IID sampling q and recording the resultant private utility values. After using those samples to form an estimate of the gradient for each agent, we update the agents' distributions accordingly. We then start another block of IID sampling to generate estimates of the next gradients.

The algorithm is provided in the Algorithm 1[3][4].

Algorithm 1 Gradient Descent on the Lagrangian

```
while System has not converge do
    create L Monte Carlo samples
    {(i.e. samples the probability distribution of each agents)}
    for each of the L samples do
        compute the world utility G
        compute the reward of each coordinate (Team Game, AU, WAU...)
    end for
    for each of the N coordinates do
        compute the component of the gradient
        update the probability distribution
    end for
end while
```

3.1. Semi-Coordinates transformation

Let assume we are a system designer of a MAS. How do we de£ne the joint-strategies of the agents? Let us present a trivial example. Let us consider two agents, R and C, which have two different actions, denoted 0 and 1. The four different states are distinct with four different payoffs. In this context, what we call semi-coordinates are the different possible joint-strategies we can de£ne: $(C, R) \mapsto state$. The choice of the mapping, as we shall see, may play an important role.

Formally, this is expressed via the standard rule for transforming probabilities,

$$P(z) = \int dx P(x) \delta(z - \zeta(x)),$$

where $\zeta(.)$ is the mapping from x to z. To see what this rule means geometrically, let $\mathcal{P}$ be the space of all distributions (product or otherwise) over z's. Let $\mathcal{Q}$ be the space of all product distributions over x. Let $\zeta(\mathcal{Q})$ be its image in $\mathcal{P}$. Then by changing $\zeta(.)$, we change that image; different choices of $\zeta(.)$ will result in different manifolds $\zeta(\mathcal{Q})$.

In £gure 1, we present two different semi-coordinates: z consists of the possible joint strategies, labelled $(1,1), (1,2), (2,1)$ and $(2,2)$. Have the space of possible x equal the space of possible z, and choose $\zeta(1,1) = (1,1)$, $\zeta(1,2) = (2,2)$, $\zeta(2,1) = (2,1)$, and $\zeta(2,2) = (1,2)$. Say that q is given by $q_1(x_1 = 1) = q_2(x_2 = 1) = 2/3$. Then the distribution over joint-strategies z is $P(z = (1,1)) = P(x = (1,1)) = 4/9$, $P(z = (2,1)) = P(z = (2,2)) = 2/9$, $P(z = (1,2)) = 1/9$. So $P(z) \neq P(z_1)P(z_2)$; the strategies of the players are statistically coupled.

$$\begin{array}{ccc} & \multicolumn{2}{c}{R} \\ & 1 & 2 \\ C\ 1 & (1,1) & (1,2) \\ 2 & (2,1) & (2,2) \end{array} \quad \mapsto \quad \begin{array}{ccc} & \multicolumn{2}{c}{R} \\ & 1 & 2 \\ C\ 1 & (1,1) & (2,2) \\ 2 & (2,1) & (1,2) \end{array}$$

Figure 1. Two different semi-coordinates in a 2 by 2 game

There are different goals associated with the idea of semi-coordinate transformation. One is to allow us to £nd a good coordinate system to start with or to be re-used in later searches. A search of a good coordinate system would then be seen as a prepocessing stage before solving a problem. Another is that, in the context of a descent, the system is likely to fall into a local minimum. Changing coordinates might allow to escape from it. Assume the system reached a local minimum, and let us assume we perform a coordinate transformation: in the new coordinate system, the landscape of the function will change shape, but the system is still at the same position. We hope that this change will create a new direction to keep on descending. By iterating the process, we believe that we should reach the global minimum.

3.2. Example

Let assume for now that we only have two different payoffs, a high value (H) and a low value (L). The ultimate goal of the agent is to maximize the payoff. We present in Figure 2 two different game matrices corresponding to two different coordinate systems for the same problem.

Matrix 1

0	H	L
1	L	H
actions	0	1

Matrix 2

0	L	L
1	H	H
actions	0	1

Figure 2. two different coordinate systems yielding to two different games

3 The stopping criteria is for now only based on the change of the probability distribution: if the change in probability falls under a £xed threshold, we stop the algorithm.

4 the step size is not held £xed. We perform a line search on the step size to ensure that L decreases, if it does not, we reduce the step size.

Using Matrix 1, a reinforcement learning algorithm will converge to a £xed strategy to get H. But if we change coordinate and consider Matrix 2, the problem become easier to solve because the effort of the coordination is less. Moreover, it is possible to improve the entropy term, if the agents try to maximize the sum of the payoff and the entropy, R converges to play action 1 and C converges to a mixed strategy $\frac{1}{2}, \frac{1}{2}$. Increasing the entropy may enable us to be more ¤exible.

3.3. Extension from a two players game

Assume now that we have N coordinates with binary actions. Recall we need to form the mapping, i.e. how do we set the joint actions of the agents. Searching over all possible transformations is not feasible (2^N! possibilities). We have not developed any theory to £nd a good coordinate system.

In this paper, we explore some preliminary techniques to make changes between two coordinates during the gradient descent proposed in Algorithm 1. We are going to try to make things better between two coordinates, hoping that it will not turns things worse with the other coordinates. This can be seen as two agents trying to collaborate by exchanging information in order to improve the system.

Such coupling of the players' strategies can be viewed as a manifestation of sets of potential binding contracts. To illustrate this return to our two player example from Figure 1. Each possible value of a component x_i determines a pair of possible joint strategies. For example, setting $x_1 = 1$ means the possible joint strategies are $(1, 1)$ and $(2, 2)$. Accordingly such a value of x_i can be viewed as a set of preferred binding contracts. The value of the other components of x determines which contract is accepted; it is the intersection of the preferred contracts offered by all the components of x that determines what single contract is selected. Continuing with our example, given that $x_1 = 1$, whether the joint-strategy is $(1, 1)$ or $(2, 2)$ (the two options offered by x_1) is determined by the value of x_2.

Binding contracts are a central component of cooperative game theory. In this sense, semi-coordinate transformations can be viewed as a way to convert noncooperative game theory into a form of cooperative game theory.

While the distribution over x uniquely sets the distribution over z, the reverse is not true. However so long as our Lagrangian directly concerns the distribution over x rather than the distribution over z, by minimizing that Lagrangian we set a distribution over z. In this way we can minimize a Lagrangian involving product distributions, even though the the associated distribution in the ultimate space of interest is not a product distribution.

In practice, when we change to change the coordinate system, we £rst choose (randomly) two coordinates. Then, we decide on the de£nition of the joint move space of these two agents. In other words, if each agents have p actions, we need to allocate the p^2 de£nition of the joint actions. For example, in the case where the actions are binary, the four joint actions labelled a, b, c and d. A shuf¤e is presented in the Figure 3. Each agent is keeping its probability distribution over its action space, hence the probabilities that each joint actions occur has changed. In the example, the probability of the joint action b does not change, whereas the probability of joint action a, which was $p_R(0) * p_C(0)$ is now $p_R(1) * p_C(1)$.

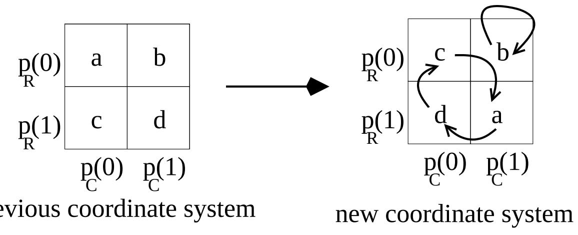

Figure 3. Example of the de£nition of the joint action space of the two agents taking part in the transformation.

We now present the different way we considered to choose the transformation.

local gradient descent We assume in this part that the size of the action space is not too large. For each different de£nition of the joint move, we can perform a 'local' gradient descent where we update only the probability of the two agents concerned in the transformation. The de£nition chosen will be the one with the best value of the world utility. We will experiment the possibility to re-use or not the new probabilities of the agents. Since only two agents are concerned, this should be very fast, but we need to perform a gradient descent for all possibles de£nition, which is possible only if the action space is small.

Based on the value of the expected G From the Monte Carlo simulation, we can compute an estimate of the expected World Utility for the different joint actions. We can get the the probability distribution of the two agents. This probability distribution remaining £xed during the transformation, one can compute the best allocation of the joint moves to optimize the value of expected world utility G. This will ensure that we reach a better value of the world utility, also, this can be done very fast.

For example, in Figure 4, in the original coordinate system, the number in the matrix are the expected value of the World utility for each joint move. The expected world utility is 2.02. If we re-assign the action space, the expected world utility can be improved to 2.26.

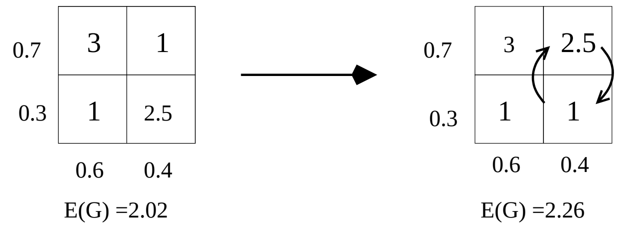

Figure 4.

We present in Algorithm 2 the algorithm we will use in the experiments. Note that we will perform a shuf¤e each time the system is next to convergence, i.e. when the system reaches a local minimum. We also ran experiments where we perform transformation periodically.

Algorithm 2 Gradient Descent with shuf¤e

```
while System has not converge do
    create L Monte Carlo samples
    for each of the L samples do
        compute the world utility G
        compute the reward of each coordinate (Team Game, AU...)
    end for
    for each of the N coordinates do
        compute the component of the gradient
        update the probability distribution
    end for
    if change in the probability $\leq$ threshold then
        choose a pair of agents
        choose a shuf¤e
        perform the coordinate transformation
    end if
end while
```

4. Experimental Results

We made some experiments in a simple coordination game where the agents are in a ring, and they must pick an action which must be opposite to their neighbors. The agent are playing in a team game. They do not get to know what is the world utility function and they do not get to observe the actions of the other players. The agents that are allowed to change their coordinate are necessarily neighbors.

All the curve presented are averaged over several runs. In Figure 5, we present results where we used a local gradient descent, and we tried out the re-use the new probabilities or not. The ring is composed by 20 agents, and the temperature is moderate (which means that the agents are not fully rational). The performance compared to a simple gradient descent is important. In Figure 6, we present results where the agents are using either a random shuf¤e, or a shuf¤e based on the expected G. For these experiments, the transformation occurred every 10 runs until iteration 100. In his case the number of agents is 50. Surprisingly, the random shuf¤e has some bene£t over not doing any transformation. It seems that the system has changed suf£ciently so that the system can reach a better minimum.

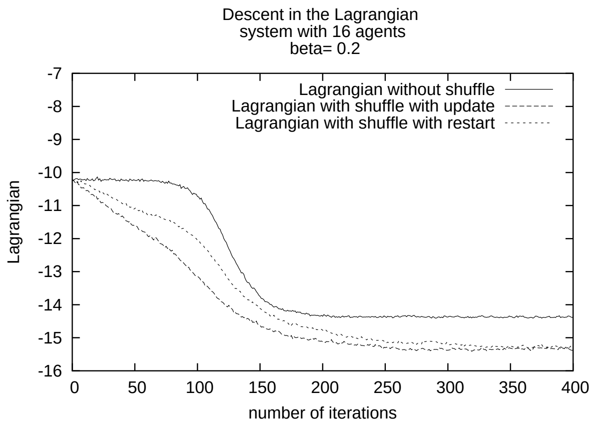

Figure 5.

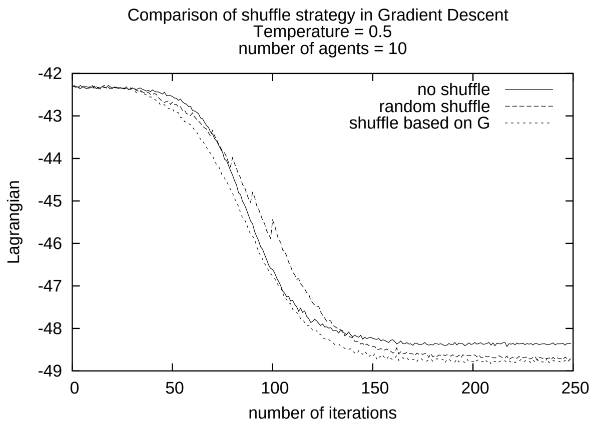

Figure 6.

5. future work

We are currently investigating other criteria to decide on the shuf¤e to perform. In particular, we are trying to understand whether the gradient information can be used. Intuitively, if the system get stuck in a local minimum, we need to look for transformation that provides a new possibility to go downhill. Hence, we are investigating transformations that yield some improvement for the expected Lagrangian and have a potentially important gradient.

Also, we are investigating ways to apply transformation on a larger set of agents. In the current implementation, we are making only one coordinate transformation between two agents. In large systems, it might be dif£cult to see the improvement made by such a local changes. We are investigating ways to make multiple local change in one iteration. Another question is about when does a shuf¤e need to be performed.

6. conclusion

Product Distribution (PD) theory is a recently introduced broad framework for analyzing, controlling, and optimizing distributed systems [8, 9, 10]. Here we investigate PD theory's use for adaptive, distributed control of a MAS. Typically such control is done by having each agent run its own reinforcement learning algorithm [3, 12, 13, 11].

In this approach the utility function of each agent is based on the world utility $G(x)$ mapping the joint move of the agents, $x \in X$, to the performance of the overall system. However in practice the agents in a MAS are bounded rational. Moreover the equilibrium they reach will typically involve mixed strategies rather than pure strategies, i.e., they don't settle on a single point x optimizing $G(x)$. This suggests formulating an approach that explicitly accounts for the bounded rational, mixed strategy character of the agents.

PD theory directly addresses these issues by casting the control problem as one of minimizing a Lagrangian of the joint probability distribution of the agents. This allows the equilibrium to be found using gradient descent techniques. In PD theory, such gradient descent can be done in a distributed manner.

We present experiments where we perform semi-coordinate transformation, that is changing the de£nition of the joint strategies of the agent during the gradient descent. The experimental results shows that these transformations are helpful to improve the speed of convergence and improve the quality of the equilibrium found by escaping local minima. It is interesting to notice that, by making several local changes in the system, we can affect the performance of the overall system. These preliminary results are encouraging.

Acknowledgments: I would like to thank Stefan Bieniawski and Chiu Fan Lee for helpful discussion.

References

[1] N. Antoine, S. Bieniawski, I. Kroo, and D. H. Wolpert. Fleet assignment using collective intelligence. In *Proceedings of 42nd Aerospace Sciences Meeting*, 2004. AIAA-2004-0622.

[2] S. Bieniawski and D. H. Wolpert. Adaptive, distributed control of constrained multi-agent systems. 2004. Submitted to AAMAS 04.

[3] R. H. Crites and A. G. Barto. Improving elevator performance using reinforcement learning. In D. S. Touretzky, M. C. Mozer, and M. E. Hasselmo, editors, *Advances in Neural Information Processing Systems - 8*, pages 1017–1023. MIT Press, 1996.

[4] D. Fudenberg and D. K. Levine. *The Theory of Learning in Games*. MIT Press, Cambridge, MA, 1998.

[5] D. Fudenberg and J. Tirole. *Game Theory*. MIT Press, Cambridge, MA, 1991.

[6] D. Mackay. *Information theory, inference, and learning algorithms*. Cambridge University Press, 2003.

[7] S. B. W. Macready and D. Wolpert. Adaptive multi-agent systems for constrained optimization. 2004. Submitted to AAAI 04.

[8] D. H. Wolpert. Factoring a canonical ensemble. 2003. cond-mat/0307630.

[9] D. H. Wolpert. Bounded rational games, information theory, and statistical physics. In D. Braha and Y. Bar-Yam, editors, *Complex Engineering Systems*, 2004.

[10] D. H. Wolpert. Generalizing mean £eld theory for distributed optimization and control. 2004. Submitted.

[11] D. H. Wolpert and K. Tumer. Optimal payoff functions for members of collectives. *Advances in Complex Systems*, 4(2/3):265–279, 2001.

[12] D. H. Wolpert and K. Tumer. Collective intelligence, data routing and braess' paradox. *Journal of Arti£cial Intelligence Research*, 2002.

[13] D. H. Wolpert, K. Wheeler, and K. Tumer. Collective intelligence for control of distributed dynamical systems. *Europhysics Letters*, 49(6), March 2000.